Logging with Perl
Tom Hukins
Logging with Perl
(because Robbie told me to)
Sorry for the boring slides
(I was tired and I have no artistic ability)
Why I Like Logs

- Debugging information during development
- Watch what code does while it runs
- Make my systems administrator happy
- Self-documenting code
Log::Dispatch

- Levels: importance
- Dispatchers: where to send messages
Using Log::Dispatch

my $dispatcher = Log::Dispatch->new();
$dispatcher->add( Log::Dispatch::File->new(
    name => 'file1',
    min_level => 'debug',
    filename => 'logfile',
),
);

$dispatcher->log(
    level => 'error',
    message => 'I need food'
);
Log::Dispatch Dispatchers

- File
- Syslog
- Email
- Screen
- Atom feed
- DBI
- Jabber
- Windows event log

...and more
Log::Log4perl

- Wraps Log::Dispatch
- Adds more features
  - categories (different parts of the system)
  - powerful configuration
  - complexity
- Based on Java's log4j
Log::Log4perl Levels

- FATAL  
  most serious
- ERROR
- WARN
- INFO
- DEBUG
- TRACE  
  least serious
Configuring Log::Log4perl

log4perl.logger = DEBUG, File
log4perl.logger.Stomach = WARN, Mailer
Configuring Log::Log4perl

log4perl.appender.File = \Log::Log4perl::Appender::File

log4perl.appender.File.filename = /path/to/log

log4perl.appender.File.mode = append

log4perl.appender.File.layout = \Log::Log4perl::Layout::PatternLayout

log4perl.appender.File.layout.ConversionPattern = \%P [%d] %F %L %c - %m%n
Using Log::Log4perl

Log::Log4perl::init('config/log.conf', 30);

# later on in your code
my $log = Log::Log4perl->get_logger('Stomach');
if ($is_hungry) {
    $log->error('I need food');
}
# Don't log messages on Saturday or Sunday

log4perl.filter.SkipSMS = sub { 
    my @date = time; 
    ($date[6] == 6 || $date[6] == 0) ? 0 : 1 
}
Other Logging Modules

- They exist
- I probably wouldn't use them
- I definitely wouldn't write my own